PATENT ABSTRACTS OF JAPAN

(11)Publication number : 09-292991
(43)Date of publication of application : 11.11.1997

(51)Int.Cl. G06F 9/38

(21)Application number : 08-108814
(22)Date of filing : 30.04.1996

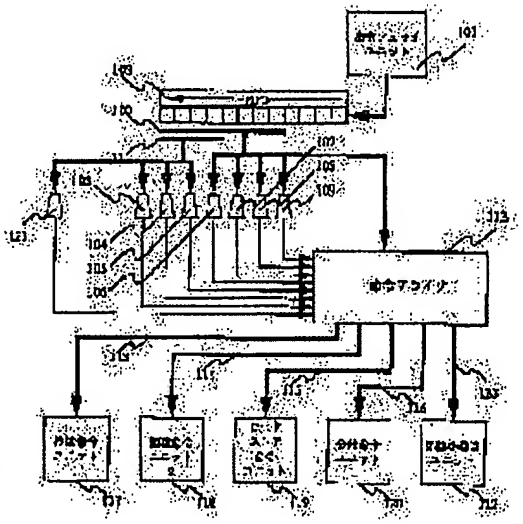
(71)Applicant : NEC CORP
(72)Inventor : OKAMURA ATSUSHI

(54) INSTRUCTION PROCESSING METHOD AND INSTRUCTION PROCESSOR

(57)Abstract:

PROBLEM TO BE SOLVED: To increase the issuing number of simultaneous instruction in an arithmetic processing unit by executing a basic instruction set by one by one instruction when the fetched instruction does not match with a parallel instruction set.

SOLUTION: Two integer instruction units 117 and 118, a load storage instruction unit 119, a floating point unit 122 and a branch instruction unit 120 exist in a processing unit. The instruction set which is the instruction given to the arithmetic processing unit and which is equivalent to an arithmetic processing unit that cannot be dissolved much shorter is set to be basic instruction set. The instruction set obtained by combining the plural basic instruction sets is defined as the parallel instruction set. When the instruction fetched by an instruction fetch mechanism is matched with the parallel instruction set, the parallel instruction sets are simultaneously issued and processed. When the fetched instruction is not matched with the parallel instruction set, the basic instruction set is individually executed one by one.



(19)日本国特許庁（ＪＰ）　(12)公開特許公報（A）　(11)特許出願公開番号

特開平9－292991

(43)公開日　平成９年(1997)11月11日

(51)Int.Cl.[6]	識別記号	庁内整理番号	ＦＩ		技術表示箇所
Ｇ０６Ｆ　9/38	３１０		Ｇ０６Ｆ　9/38	３１０Ｈ	
				３１０Ａ	

審査請求　有　　請求項の数12　ＯＬ　（全 10 頁）

(21)出願番号　特願平8－108814

(22)出願日　平成８年(1996)４月30日

(71)出願人　000004237
日本電気株式会社
東京都港区芝五丁目７番１号

(72)発明者　岡村　淳
東京都港区芝五丁目７番１号　日本電気株式会社内

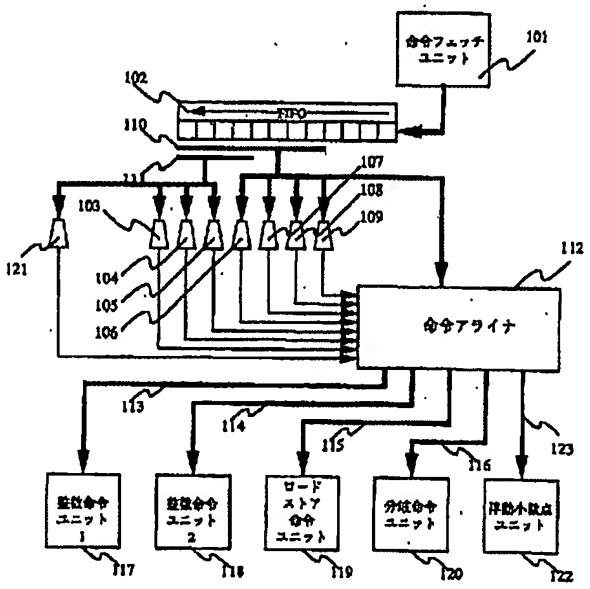
(74)代理人　弁理士　鈴木　章夫

(54)【発明の名称】　命令処理方法及び命令処理装置

(57)【要約】

【課題】　可変長命令セットをもつ命令処理装置でより多くの命令を並列フェッチし、デコードすることを可能とする。

【解決手段】　演算処理装置に与えられる命令で、それ以上短く分解する事のできない演算処理単位に相当する命令セットを基本命令セットとすると、この基本命令セットを複数組み合わせた命令セットを並列命令セットとして定義し、命令フェッチ機構によってフェッチされた命令が、前記並列命令セットに一致した場合は、並列命令セットを同時に発行，処理し、フェッチされた命令が、前記並列命令セットに一致しない場合は、前記基本命令セットを単独で１命令ずつ実行する。



1

【特許請求の範囲】

【請求項１】　演算処理装置に与えられる命令で、それ以上短く分解する事のできない演算処理単位に相当する命令セットを基本命令セットとすると、前記基本命令セットを複数組み合わせた命令セットを並列命令セットとして定義し、命令フェッチ機構によってフェッチされた命令が、前記並列命令セットに一致した場合は、並列命令セットを同時に発行，処理し、前記フェッチされた命令が、前記並列命令セットに一致しない場合は、前記基本命令セットを単独で１命令ずつ実行することを特徴とする命令処理方法。

【請求項２】　前記基本命令セットを、一定の順序で複数組み合わせて任意の長さにした命令セットを並列命令セットとして定義する請求項１の命令処理方法。

【請求項３】　前記基本命令セットを、一定の順序で複数組み合わせて演算処理装置特有のワード長の任意倍の長さにした命令セットを並列命令セットとして定義する請求項１の命令処理方法。

【請求項４】　前記フェッチされた命令が、前記並列命令セットに一致しない場合は、前記基本命令セットとして、スーパースケーラ型複数命令デコード論理を用いて複数発行を行う請求項１または３の命令処理方法。

【請求項５】　前記基本命令セットを、その命令を処理するユニットによって複数のグループに分割し、前記グループの何れか１つに属する種類の命令のみを置くことのできる命令フィールドをスロットとした場合、前記スロットを複数並べた命令セットを並列命令セットとして定義する請求項１ないし４のいずれかの命令処理方法。

【請求項６】　前記分割したグループの何れか１つに属する種類の命令のみを置くことのできる命令フィールドをスロットとした場合、前記スロットを一定の順序で複数組み合わせて演算処理装置特有のワード長の任意倍の長さにした命令セットを並列命令セットとして定義する請求項５の命令処理方法。

【請求項７】　前記基本命令セットを、その命令を処理するユニット及び命令の長さによってグループに分割する請求項５または６の命令処理方法。

【請求項８】　前記スロットを任意の順序で並べた命令列を並列命令セットとすると、単一の前記並列命令セットを構成する複数の前記スロットに含まれる基本命令も、必ず単独で処理する請求項７の命令処理方法。

【請求項９】　演算処理装置に与えられる命令で、それ以上短く分解する事のできない演算処理単位に相当する命令セットを基本命令セットとすると、前記基本命令セットを複数組み合わせた命令セットを並列命令セットとして定義し、命令フェッチ機構によってフェッチされた命令が、前記並列命令セットに一致した場合は、並列命令セットを同時に発行，処理し、前記フェッチされた命令が、前記並列命令セットに一致しない場合は、前記基本命令セットを単独で１命令ずつ実行することを特徴とする命令処理装置。

2

【請求項１０】　メモリ装置から命令をフェッチし、読み込んだメモリを順に出力バスに出力する命令フェッチ装置と、入力バスを有し、前記入力バスから一定の幅の命令列を挿入し、挿入されたのと同じ順序で、出力バスを通して、任意の幅の命令列を読み出すことのできるＦＩＦＯメモリ装置と、１つ以上の命令をチェックする命令列チェック装置と、発行された命令を処理、実行する１つ以上の命令実行装置と、発行する命令列を入力するバスを有し、入力バスを通して入力された命令列を一定のフォーマットに整形し、複数の出力バスに接続する前記複数の実行ユニットに分配する命令分配アライン装置を有し、前記命令フェッチ装置の命令出力バスを前記ＦＩＦＯメモリ装置の入力バスに接続し、前記命令デコード装置を前記ＦＩＦＯメモリ装置の前記出力バスから前記命令列チェック装置の入力バス及び、命令分配アライン装置の入力バスに接続し、命令分配アライン装置の出力バスに、前記複数の命令実行装置の入力に接続し、前記命令フェッチ装置は、実行すべき命令列をフェッチし、先に実行すべき命令から順にバスを介して前記ＦＩＦＯメモリ装置に投入し、前記命令チェック装置は、前記ＦＩＦＯメモリ装置に挿入された命令列を前記ＦＩＦＯメモリ装置の命令の出口側から命令で２つ以上サイズを含む幅でチェックし、前記２つ以上の命令が命令チェック装置に固有の命令列を有しているならば真を出力し、異なっているならば偽を出力し、命令分配アライン装置は、複数の命令チェック装置の何れかが真を出力している場合は、前記ＦＩＦＯメモリ装置から読み出された命令列を、命令チェック装置が真を返した形式の命令処理ネットワークを用いて分配整列し、前記命令列の含む複数の命令を同時に前記各実行装置に前記バスを通して転送し、前記複数の命令チェック装置のすべてが偽を出力した場合は、前記ＦＩＦＯメモリ装置から１命令分の命令列を取り込み、前記１命令を前記各命令実行装置の何れかに転送し実行することを特徴とする命令処理装置。

【請求項１１】　前記複数の命令チェック装置のすべてが偽を出力した場合は、前記ＦＩＦＯメモリ装置から複数命令分の命令列を取り込み、前記命令列を順に解析し、対応する前記各命令実行装置の何れかに転送し実行する請求項１０の命令処理装置。

【請求項１２】　演算処理装置に与えられる命令で、それ以上短く分解することのできない演算処理単位に相当する命令セットを基本命令セットとし、前記基本命令セットを、一定の順序で複数組み合わせて任意の長さにした命令セットを並列命令セットとすれば、演算処理装置の実行する命令セットを解析する場合に、前記並列命令セットを解析する装置と、前記基本命令セットを解析する装置を有する請求項９ないし１１のいずれかの命令処理装置。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は命令処理方法及び命令処理装置に関する。

【0002】

【従来の技術】一般に、演算処理装置の高性能化の技術として、ＶＬＩＷ方式やスーパースケーラ方式がある。いずれの方式も、命令を並列に実行することによって、１クロック当たりに処理可能な命令数を増やし、高性能化を図る技術である。例えば、ＶＬＩＷ方式は、通常の演算処理装置に与えられる命令を小命令とし、幾つかの決まった数の前記小命令を一定の制限をもってつなげた命令群を１大命令とする。そして、前記大命令を一度に読み込み、前記大命令中の小命令を並列に処理している。ＶＬＩＷのセマンティックは、同一の大命令に含まれる小命令は同時に処理されることである。

【0003】図４にＶＬＩＷ命令のフォーマットの例を示す。図４の３２バイト分４０７を１命令として読み込み同時に実行する。前記３２バイトの命令を大命令とする。前記大命令は幾つかの小命令に分割され、それぞれの小命令が固有の命令フォーマットを持つ。同図では大命令は６個の小命令に分割され、小命令は左からロードストア命令４０１、ロードストア命令４０２、ＡＬＵ命令４０３、ＡＬＵ命令４０４、ＩＭＭ命令４０５、分岐命令４０６が置かれている。これらの並べた命令は同時に実行されることが保証されている。ここで、ロードストア命令はメモリから演算処理装置の内部汎用レジスタにデータを転送する命令であり、ＡＬＵ命令は前記内部汎用レジスタ間の演算、ＩＭＭ命令はオペランドの値を汎用レジスタにセットする命令、分岐命令は次に実行する命令の番地を変更するものである。

【0004】このようなＶＬＩＷ方式は、命令の並びかた、小命令のフィールド長が演算処理装置固有のものになるため、互換性が非常に乏しいという問題がある。すなわち、前記大命令は一度に読み込まれ処理されるため、命令列は１つの前記大命令に含まれている前記小命令が無条件に同時に実行できるように並べなければならない。もし、前記小命令間の依存関係によって小命令が同時に実行できない場合は、ｎｏｐ（ノーオペレーション）命令を挿入することで、一度に実行されてしまう前記大命令中の小命令の依存関係を保証する。以上のように、ＶＬＩＷでは、命令コード自体が並列に実行できるように並べられる事を特徴とする。これをスタティックスケジューリングという。

【0005】一方、スーパースケーラ方式は、従来との互換性をとった上で命令を並列に実行できるようにする技術である。通常の演算処理装置の命令コードは、アドレスの小さなものから１つずつ実行するような規則で並べられている。このため、、前にある命令を実行し終えてから次の命令を実行する。前にある命令の実行結果を後で実行する命令が使用する場合がある為である。前記結果の使用を依存関係と呼ぶ。すべての命令が前記依存関係を有しているわけではないため、依存関係のない命令は、先に実行してしまっても問題は発生しない。前記したＶＬＩＷ方式では、この依存関係を持たない命令をコンパイル時に前記大命令として並べ並列処理を実現しているのに対し、スーパースケーラ方式は、命令列を実行すべき順（プログラムオーダ）に取り込み解析し、前記依存関係が無い命令を選びだし、先に処理をすることができる命令の実行を前倒しして並列実行を実現する。このような並列命令処理方式としては、特開平２－１３０６３５号公報に記載の複数命令同時処理方式があげられる。この方式では、複数個の命令を同時に読み出し、読み出された命令を複数個のデコーダで解析し、依存関係が無い場合のみ、複数の実行手段で並列に実行する処理方式が用いられている。

【0006】また、前記方式よりも一段階進んだスーパースケーラ方式の一例を図５に示す。これは、ＩＢＭ社のＩＢＭ３６０／９１で使用されている方法である。フェッチユニット５０１はメモリから命令をフェッチしデコードする。デコード結果によって命令を対応する演算器の種類毎に分割し、もしリザベーションステーション５０４，５０８に空きがあれば当該命令を発行する。更に、ソースオペランドがレジスタファイル５０２に存在すれば、それもリザベーションステーション５０４，５０８に格納する。もし空きが無いならば、フェッチユニット５０１は、命令を発行せず、空きが生ずるまで当該命令を発行は停止する。リザベーションステーション５０４，５０８中に発行された命令は、ソースオペランドがそろっていない場合、共通データバス５０６を監視して、対応するオペランドが前記共通データバス上に現れるのを待つ。ソースオペランドが前記共通データバス５０６上に現れたら、その値をソースレジスタ値バス５０３経由で前記リザベーションステーション５０４，５０８中に書き込む。

【0007】リザベーションステーション５０４，５０８中の命令は、総てのソースオペランドが揃ったら、その命令を演算ユニット５０５，５０９に送る。演算ユニット５０５，５０９は実行を完了したら、実行結果を共通データバス５０６経由でレジスタファイル５０２に書き込むと同時に、当該結果を待っているかもしれないリザベーションステーションに送る。以上のような機構によって、オペランドが有効でない間、命令はリザベーションステーション中でまたされ、実行できる命令から順に優先して実行する機能を提供できる。

【0008】ところで、従来のＣＩＳＣ型命令セットでは、命令コードの静的なサイズを小さくするために、可変長命令セットが用いられている。前記、可変長命令セットは、命令によって命令フォーマットの語長が変化する命令セットである。出現頻度の高い命令を短く、出現

頻度の低い命令を長いフォーマットにすることで、プログラムの静的なコードサイズを小さくする事ができる。このような可変長命令セットの命令コードを並列実行するために、複数命令をデコードする手順について説明する。可変長命令の命令セットをスーパースケーラ方式で、実現する場合の難しさは命令フェッチ機構にある。スーパースケーラ方式では、一度に実行ユニットに送ることが可能な命令数が多ければ多いほど並列実行の可能性も増加する。可変長命令セットを持つアーキテクチャの場合は、複数命令フェッチをする場合に命令の解析手順が複雑になってしまう。

【０００９】現在のＲＩＳＣ型命令セットでは、命令は３２ビット長で更に４バイトアラインされている。このため、命令の始まりの位置は、完全に固定しており、命令の解析は、この固定位置から行うことで、複数命令の並列解析が可能になる。それに対して、可変長命令の場合は、始まりの命令の位置だけが明確にわかっており、２番目の命令の始まりの位置を見つけるためには、始めの命令を解析し、始めの命令の命令サイズを特定しなければならない。つまり、可変長命令の場合は、２番目以降の命令の始まりの位置が不明確で、前に実行される命令から順に解析しなければならない。解析しなければならない命令数が増大するに従って、命令の始めを見つける論理は積み重なってしまい、遅延時間が増大してしまうため速いクロックで動作する場合、多くの命令をデコードするのは困難になる。

【００１０】実際に高速動作させた場合は、順番にデコードしていたのでは１クロックでデコードできる命令の数が非常に限られてくる。この命令のデコードを高速化するため、命令をキャッシュに書き込む際にプリデコードを行い始めから区切りを明確化しておく方法がある。これは、ＡＭＤ社のＫ５と呼ばれるマイクロプロセッサが選択した方法である。Ｋ５では、メモリから命令を取り込みオンチップ・キャッシュに書き込む時点で、命令をデコードし命令の頭に対応するバイトにマークを付けている。この方法は、非常に優れた方法であるが、キャッシュにプリデコードビットを付加しなければならないことや、キャッシュのラインと分岐先アドレスの一致が必ずしもアラインされていないため、分岐時の制御が非常に難しくなってしまうなどの弊害が発生する。

【００１１】

【発明が解決しようとする課題】命令のスーパースケーラ型並列実行をするためには１クロック中に複数の命令をデコードする必要がある。通常のＲＩＳＣ型プロセッサでは命令コードは３２ビット固定長になっており、且つ４バイトアラインされているため、複数の命令を一度に取り込み、前記複数の命令を一斉にデコードする事が可能である。しかし、可変長命令セットを処理する演算処理装置では、一度に多くの命令を取り込んでも着目している命令の種類によって、その命令以降の命令の始まりの位置が変化してしまうため、取り込んだ多くの命令を並列にデコードする事ができず、１つずつ順々にデコードしなければならない。その結果、同時実行する多数の命令を高速にデコードする事が困難となる。

【００１２】本発明の目的は、可変長命令セットを有する演算処理装置の同時命令発行の発行数を大きくすることが可能な命令処理方法と命令処理装置を提供することにある。

【００１３】

【課題を解決するための手段】本発明は、演算処理装置に与えられる命令で、それ以上短く分解する事のできない演算処理単位に相当する命令セットを基本命令セットとすると、前記基本命令セットを複数組み合わせた命令セットを並列命令セットとして定義し、命令フェッチ機構によってフェッチされた命令が、前記並列命令セットに一致した場合は、並列命令セットを同時に発行，処理し、前記フェッチされた命令が、前記並列命令セットに一致しない場合は、前記基本命令セットを単独で１命令ずつ実行することを特徴とする。

【００１４】

【発明の実施の形態】次に、本発明の実施形態を図面を参照して説明する。図１は本発明の実施形態の装置のブロック図であり、図２は本発明の実施形態で説明する演算処理装置の命令セットの命令長２０１～２０４と複数発行時の命令の並列命令形式２０５～２１１を示す。２０１～２０４の命令を以下基本命令セットと呼ぶ。前記基本命令セットは、本演算処理装置の命令セットレベルアーキテクチャが提供するそれ以上分解されないアトミックな命令である。本実施形態では、基本命令は命令の種類によって１～４バイトの命令長を持つ。図３に各基本命令セットの命令が、どの処理グループに属するかでの分類を示す。処理グループとは、当該命令がいかなる処理ユニットで処理され得るかを基準にしたグループである。

【００１５】前記処理ユニットは図１に示すように、２つの整数命令ユニット１１７，１１８と、ロードストア命令ユニット１１９と、浮動小数点ユニット１２２と、分岐命令ユニット１２０とが存在する。このため、前記処理グループは、整数演算グループ、ロードストアグループ、浮動小数点グループ、分岐グループ、その他グループの５グループに分割される。すべての基本命令は、命令の長さによる区分と命令を処理するユニットによる区分で細分化される。複数の命令を発行する場合は、図２の２０５～２１１に示すＴｙｐｅ１～７のフォーマットにフェッチされた命令が並んでいることを必要とする。このフォーマットを以下並列命令セットと呼ぶ。本実施形態では、並列命令セットは基本命令セットを組み合わせて、決まった長さになるように、かつ、その命令の前記処理グループごとに順序を決めて制限を設けた形式になっている。本実施形態では、７種類の並列命令セ

ットを定義する。

【0016】Ｔｙｐｅ1の並列命令セットは、1バイト命令の整数演算命令、1バイト命令の整数演算命令、2バイト命令のロードストア命令をつなげたものになる。また、Ｔｙｐｅ2の並列命令セットは、1バイト命令の整数演算命令、2バイト命令の浮動小数点命令又はロードストア命令、1バイト命令の整数演算命令又はＪＲ命令をつなげたものである。以下、Ｔｙｐｅ3の並列命令セットは、1バイト命令の整数演算命令、3バイト命令の整数演算命令又は分岐命令をつなげたもの、Ｔｙｐｅ4の並列命令セットは、4バイト命令のロードストア命令、1バイト命令の整数演算命令、2バイト命令の浮動小数点演算命令、1バイト命令の整数演算命令又はＪＲ（レジスタ間接分岐）命令をつなげたもの、Ｔｙｐｅ5の並列命令セットは、2バイトの整数演算命令、2バイトのロードストア命令、2バイトの浮動小数点演算命令をつなげたもの、Ｔｙｐｅ6の並列命令セットは、2バイトの整数演算命令、2バイトの整数演算命令、4バイトのロードストア命令又は分岐命令つなげたもの、Ｔｙｐｅ7の並列命令セットは、4バイトの整数演算命令、4バイトのロードストア命令又は分岐命令をつなげたものになっている。

【0017】以上のように並列に実行される命令セットは、いくつかのテンプレートをもっており、テンプレートにマッチした場合のみ複数命令が発行される。もし、命令列がテンプレートにマッチしていない場合は、命令列は、1つずつ順々に発行される。この方式を、テンプレート方式命令発行と呼ぶ。図1のブロック図は、前記テンプレート方式命令発行を行うための処理機構を示してある。命令フェッチユニット101は、メモリ装置から実行すべき命令列を取り込む。前記実行すべき命令列は、命令ＦＩＦＯ102に入れられる。この命令ＦＩＦＯ102は、先に実行すべき命令から順に向かって右側から入れられる。命令ＦＩＦＯ中の命令は命令の発行処理が行われたものから順次、左側から取り除かれ、その分だけ命令ＦＩＦＯの中の命令は左側方向にシフトしていく。ＦＩＦＯの中には、左側に行くほど先に実行すべき命令になるように保たれる。ＦＩＦＯの中に次の命令を入れられる空間ができたら、命令フェッチユニット101は新しくフェッチした命令を挿入する。

【0018】命令ＦＩＦＯ102の左側、即ちもっとも先に実行すべき命令が入っている4バイト分を4バイトサイズの命令として、並列命令チェック論理103，104，105に接続する。また、命令ＦＩＦＯメモリ102の左側から8バイトサイズの命令を、並列命令チェック論理106，107，108，109に接続する。前記並列命令チェック論理は、論理103がＴｙｐｅ1の命令のチェックをし、論理104がＴｙｐｅ2の命令のチェックをし、論理105がＴｙｐｅ3の命令の形式のチェックを受け持つ。同様に、8バイトサイズの並列命令に関しては、論理106がＴｙｐｅ4の命令の形式を、論理107がＴｙｐｅ5の命令の形式を、論理108がＴｙｐｅ6の命令の形式を、そして、論理109がＴｙｐｅ7の命令の形式をチェックする。

【0019】命令のチェック論理は、入力されたバイト列に自分がデコードすべき命令列にあっているかどうかを判定し、もし、前記制限に対応できるような命令列になっていたら命令アライナ112に対して真を出力する。図7に命令チェック論理を示す。図7は、Ｔｙｐｅ4の命令をチェックする論理をブロックで示したものである。ＦＩＦＯ102の左側から8バイトのデータを供給される。チェック機構は、703～706の論理になる。703の論理は、4バイト形式のロードストア命令をデコードし真を出力する。それ以外が来た場合は偽を出力する。以下、704は5バイト目から始まるはずの1バイト長の整数命令を、705は6バイト目から始まるはずの2バイト長の浮動小数点命令を、706は8バイト目から始まるはずの1バイト長レジスタ間接分岐命令（ＪＲ）か整数演算命令をデコードし、合っていれば真を出力する。これらすべてのデコーダが真を出力した場合に論理和回路707によってすべての真をチェックし命令がフォーマットに合っていることが確認される。

【0020】図中では、701にマッチしない命令の場合、702にマッチする命令の場合を示す。フォーマット701の命令が入力されると1バイト目にあるデコーダ703は、ロードストア命令ではないので偽を出力する。デコーダ704と706は、命令の途中をデコードしてしまうため、正しいデコード結果を得ることができない。オペランドの値如何によっては真を返すこともある。しかし、デコーダ703が偽を出力しているため出力708には偽が出力される。フォーマット702が入力された場合は、すべてのチェック回路703～706が真を出力する事になり、出力708には真が出力される。

【0021】図1で、103～109の並列命令チェック論理の何れかから真の出力がでてくれば、命令は並列タイプとして発行され、どれも真にならない場合は1命令ずつ処理する。デコーダ121はＦＩＦＯ102の最も左側の命令をデコードし命令のサイズ及び命令の処理グループを決定し、命令アライナ112にその結果を送る。図8に命令アライナ112の内部ブロックを示す。本実施形態の命令アライナはすべての命令列を処理できるようにした場合に比べて非常に簡単である。本実施形態では、7タイプの命令をそれぞれ並べ替えるネットワークを実装すればよいためである。

【0022】図8では、834は図1のＦＩＦＯ102の左側から8バイト分に相当する。ＦＩＦＯに投入された命令の中で、最も始めに発行すべき8バイトを並べ替えることになる。同図中で、809は命令が何れのテンプレートにもマッチしない場合のアライナである。図1

のデコーダ１２１で命令のサイズをデコードし、そのサイズに応じて命令アライナ８０９でアライン処理を行う。１命令だけをデコードする回路になるため、従来のパイプライン型と同様な構成で実現できる。図８で、８１０～８１８は命令のバイトを接続し、一定のフォーマットに整形する回路である。また、８１９から８２３は、複数命令発行時の各ユニットに発行する命令を選択するセレクタである。８２４から８２８は、複数命令が発行できない場合に、単独な命令として処理する場合に、単独命令を整形するユニット８０９からの出力と選択を行うセレクタである。

【００２３】図８中では、８１０は２バイト命令を整形する回路になる。図２からも明らかであるが、テンプレートにマッチする限り、命令の第１バイト目と命令の第２バイト目は、２バイト形式の整数命令しか現れない。このため、整形回路８１０の出力は整数演算ユニット１に投入する命令を選択するためのセレクタ８１９に接続される。セレクタ８１９では、Ｔｙｐｅ５か又はＴｙｐｅ６の命令がデコーダで判定された場合の処理の為に整形回路８１９の出力を整数ユニット１に対して接続する。

【００２４】さらに、図８の８１１は、ＦＩＦＯの第１バイト目から第４バイト目までの４バイト命令を整形する整形回路で、Ｔｙｐｅ４とＴｙｐｅ７で使用される。Ｔｙｐｅ４では、ロードストア命令であるため、ロードストア処理ユニット８３１に接続されるような動作を行い、Ｔｙｐｅ７が検出された場合は、整数演算ユニット８２９に接続される様に動作する。以下、８１２はＴｙｐｅ２の浮動小数点命令かロードストア命令、８１３はＴｙｐｅ５／６の整数演算又はＴｙｐｅ１のロードストア命令、８１４はＴｙｐｅ３の整数か分岐命令、８１５はＴｙｐｅ５のロードストア命令、８１６はＴｙｐｅ４の浮動小数点命令、８１７はＴｙｐｅ５の浮動小数点命令、８１８はＴｙｐｅ６／Ｔｙｐｅ７のロードストア命令か分岐命令のフォーマット整形を行う。フォーマット整形を完了した命令群は、セレクタ８１９～８２３でＴｙｐｅ毎に選択され演算処理ユニット８２９～８３３に転送される。発行される命令列は、実装される演算ユニットを考慮に入れて決定されているため、命令チェック論理でチェックされた命令列は無条件に各ユニットに発行してよい。

【００２５】以上の各実施形態から明らかにように、この命令処理方式では、クロック周波数が高くなった場合でも可変長命令セットを有する演算処理装置の同時命令発行の発行数を非常に大きくすることができる。図７にあるように、可変長命令の場合でも、注目するビットだけをデコードすれば、前から順にデコードしたのと同じ効果を得ることができ、この方法では命令の区切りを求める必要がない。そのため、図７の例で示せば、デコード回路の部分では、１命令ずつのデコードに対して、７０７の論理積ゲート分の遅延が増加するだけである。また、命令のアライン部分も図８に示すように、完全なクロスバ型から比べて簡単になっている。このため、バイト可変長命令セットを持つ演算処理装置で、完全な対象型の命令発行方式をとった場合、本発明の方式によれば、同程度の条件で非常に多くの命令の発行が可能になる。これは、可変長命令のデコードは前から順に行わなければならない決まりを取り除くことができるためである。

【００２６】

【発明の効果】以上説明したように本発明は、全ての命令を解析するのではなく、実際のプログラム中で頻出する命令の組み合わせを並列に実行される命令のテンプレートとして、前記命令の組み合わせに適合するか同化のタイプチェック機能をデコーダに組み込み、その命令のテンプレートに一致するかどうかのチェックのみを行い、テンプレートに一致した命令は同時に多くの命令を発行することができるようにし、一致しない場合は同時の発行数を減らした通常のデコーダで処理するようにすることにより、高性能な命令処理が実現できる。特に、可変長命令を複数組み合わせることで、固定長に組み直すことも可能になるため、命令フェッチ機構全体の簡略化が実現できる。

【図面の簡単な説明】

【図１】本発明にかかるテンプレート発行方式のブロック図である。

【図２】基本命令，並列命令の典型的なフォーマット図である。

【図３】基本命令のバイト数と処理ユニットの分類を示す図である。

【図４】基本的なＶＬＩＷ命令のフォーマット図である。

【図５】スーパースケーラ処理ユニットの図である。

【図６】ＦＩＦＯユニットの構成図である。

【図７】命令チェック論理を示す図である。

【図８】命令分配アライン装置を示す図である。

【符号の説明】

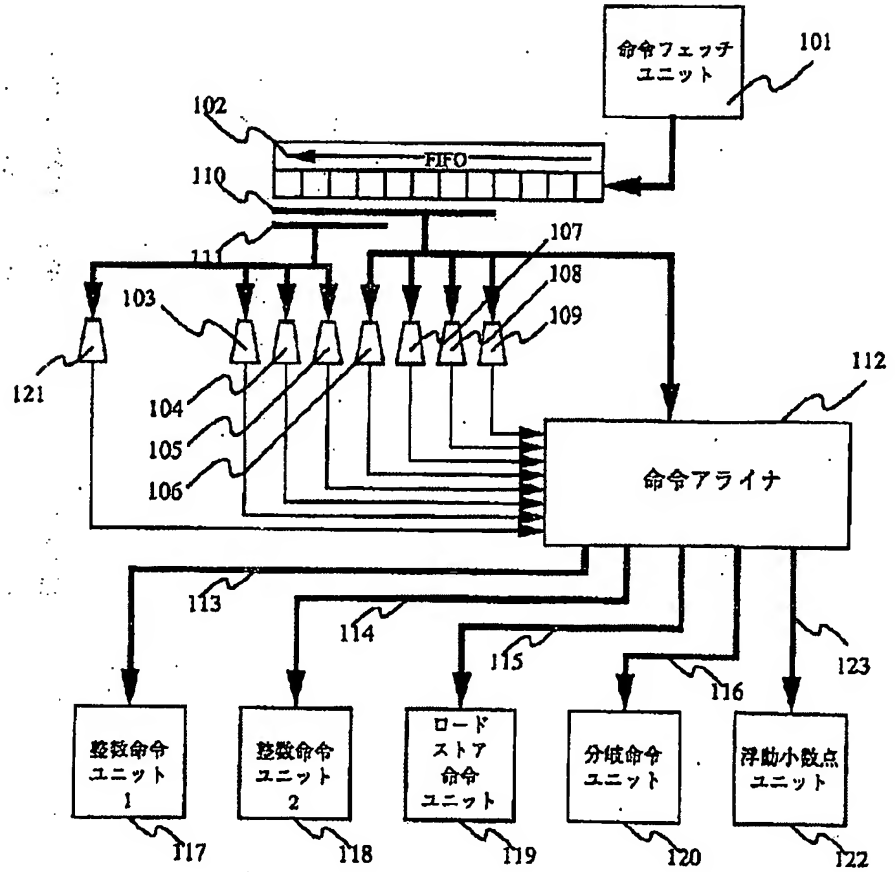
１０１　命令フェッチ機構
１０２　ＦＩＦＯ装置
１０３～１０９　命令チェック機構
１１０　８バイト命令転送バス
１１１　４バイト命令転送バス
１１２　命令分配アライン装置
１１３　整数命令発行バス
１１４　整数命令発行第２バス
１１５　ロードストア命令発行バス
１１６　分岐命令発行バス
１１７，１１８　整数演算命令実行ユニット
１１９　ロードストア命令実行ユニット
１２０　分岐命令実行ユニット

11　　12

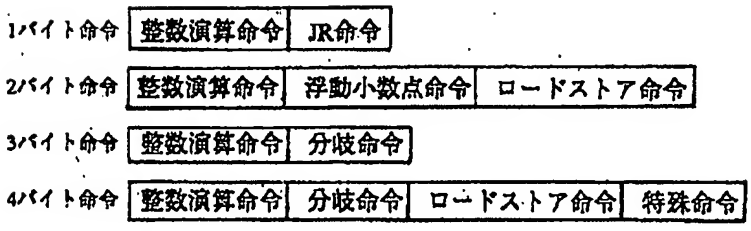
１２１　単独命令デコードユニット　　＊１２３　浮動小数点演算命令発行バス

１２２　浮動小数点命令実行ユニット　　＊

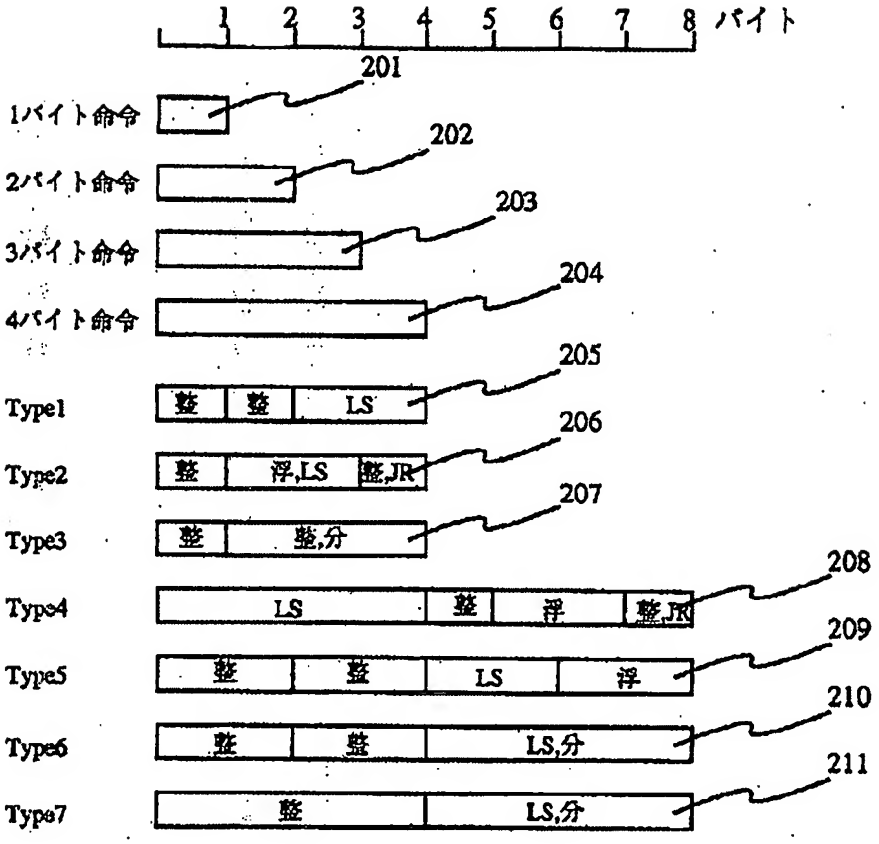
【図１】



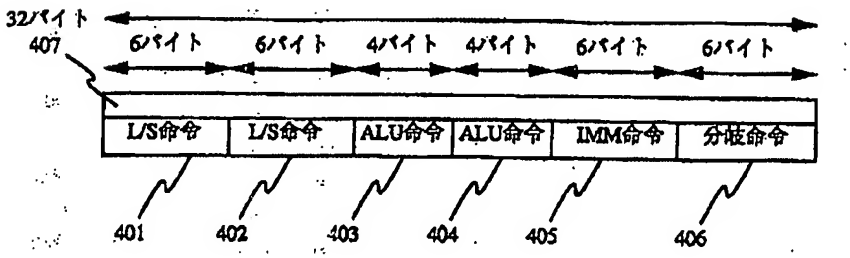
【図３】



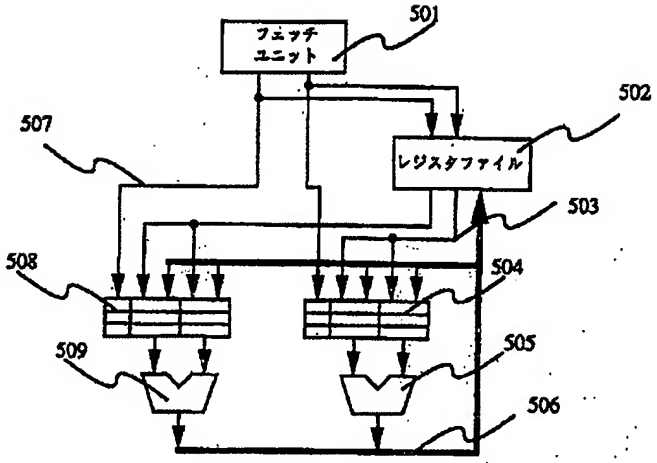
【図2】


1 2 3 4 5 6 7 8 バイト
201
1バイト命令
202
2バイト命令
203
3バイト命令
204
4バイト命令
205
Type1
整
整
LS
206
Type2
整
浮,LS
整,JR
207
Type3
整
整,分
208
Type4
LS
整
浮
整,JR
209
Type5
整
整
LS
浮
210
Type6
整
整
LS,分
211
Type7
整
LS,分

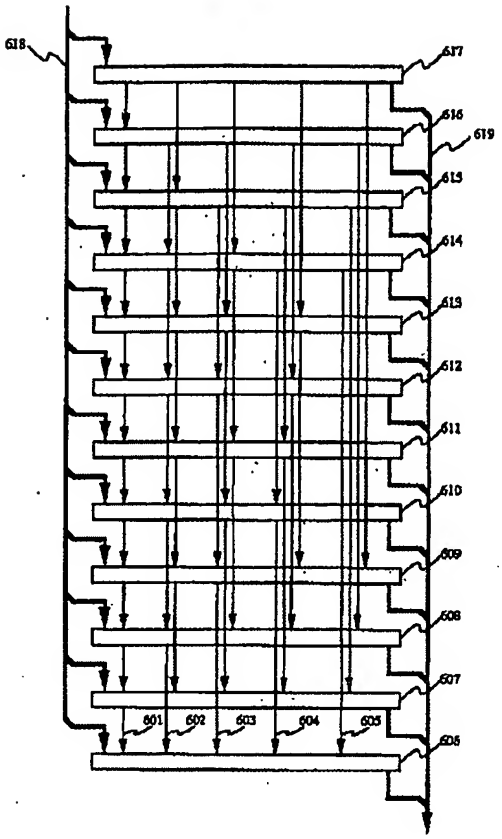
【図4】


32バイト
407
6バイト
6バイト
4バイト
4バイト
6バイト
6バイト
L/S命令
L/S命令
ALU命令
ALU命令
IMM命令
分岐命令
401
402
403
404
405
406

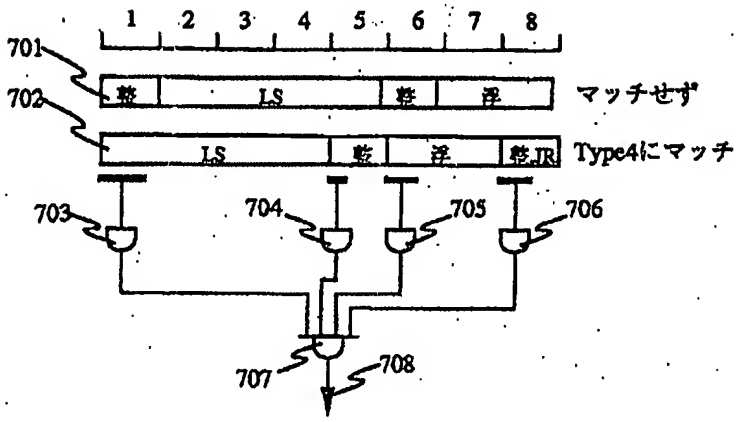
【図5】


フェッチ
ユニット
501
502
レジスタファイル
503
504
505
506
507
508
509

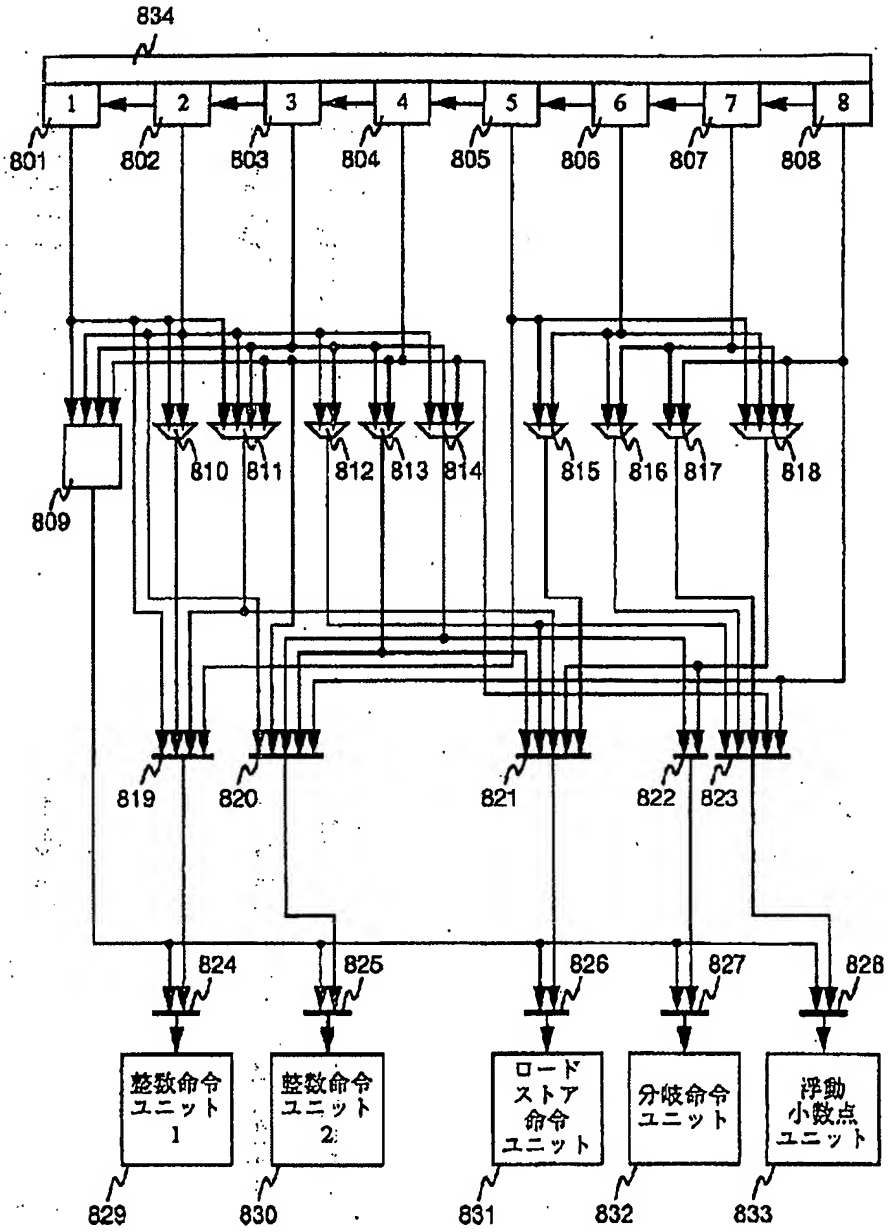
【図6】


618
617
616
619
615
614
613
612
611
610
609
608
607
601
602
603
604
605
606

【図7】


1
2
3
4
5
6
7
8
701
整
LS
整
浮
マッチせず
702
LS
整
浮
整JR
Type4にマッチ
703
704
705
706
707
708

【図8】


834
1
2
3
4
5
6
7
8
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
整数命令ユニット1
整数命令ユニット2
ロードストア命令ユニット
分岐命令ユニット
浮動小数点ユニット
829
830
831
832
833